A Compressed Dual System for Road Object Detection

Kazi Safkat Taa Seen
Department of Electrical and Computer Engineering
University of Arizona
Tucson, AZ 85721
safkat@arizona.edu

Abstract

In this study, we present a dual-backbone localize and classify approach to detect objects in thermal images. We utilize SOTA localizers and classifiers to predict and classify bounding boxes accurately. Existing approaches utilize a one-stage detector that sometimes fails to classify or detect objects due to its dependency on a singular backbone. Furthermore, we realize the necessity of such systems to be both accurate and efficient; as a result, we utilize several model compression techniques and also develop a system to deploy the model. We use the Teledyne FLIR dataset as a source of thermal imagery appropriate for road vehicles and systems. In this study, we achieved mAP values of 22.96% and 22.38% for our experiments, outperforming the normal Faster RCNN model (19.16%).

1 Introduction

The enhancement of computer vision techniques has enabled the advent of AI automated vehicles LeCun et al. (2015). Such has also been the case with object detection Girshick (2015); He et al. (2017); Liu et al. (2016). Object detection aims to detect/localize objects in the image and accurately classify each of them. Recent popular methods utilize a singular backbone Girshick (2015); Liu et al. (2016) where one is a two-stage process mainly utilizing a dense backbone - FPN (Feature proposal Networks) - to generate feature maps. These maps are then used to propose regions (containing images) using a region proposal network (RPN). The regions are then used to predict accurate bounding boxes and classify images using very small neural networks. Other singular backbones Redmon and Farhadi (2017); Lin et al. (2017) are one stage and define marks on pixels, predict bounding boxes, and then classify using smaller sub-networks. These methods are well-established and popular due to their performance efficacy, robustness, feasibility Lin et al. (2014); Everingham et al. (2010), ease-of-use, and quick inference time Wu et al. (2019).

However, networks, as such, work on features extracted on a large-scale image or the complete image containing several objects. This does not allow the backbones to capture all the intricacies of each individual object, resulting in wrong classifications in singular backbone architectures. Works have investigated enhancing classification by using different loss functions Lin et al. (2017), augmenting images and regions of interest. Also a backbone picking up details in objects might miss important object locations an image. In this study, we look to solve both classification and localization inaccuracies by utilizing two backbones, one for localization and one for classification. The two backbones take up different tasks and learn features accordingly to perform better. We understand the two backbones make the model heavy in computation and efficiency. As a result, we look into several model compression methods, such as knowledge distillation and quantization on the classification model. For knowledge distillation, we take up a teacher-free (synthetic teacher) approach and utilize focal loss with Kull-back Liebler loss instead of Cross-Entropy Loss with Kull-back Liebler Loss to tackle the class imbalance problem. We further augment imbalanced classes. We finally quantize the classification model using post-training quantization to get a faster, smaller, and more efficient

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

model. We deploy this model into our local server, where a user can upload an image and get accurate detections and classifications of objects in the image.

Another issue with current object detection studies is that they deal with RGB/Vision images. Vision images in normal conditions are susceptible to a lot of noise, such as light, glare, weather conditions, etc. We investigate the FLIR thermal dataset to tackle this problem.

2 Technical Description

This section discusses the models, training strategies, inference techniques used in this study. Figure 1 illustrates the complete architecture utilized in this study.

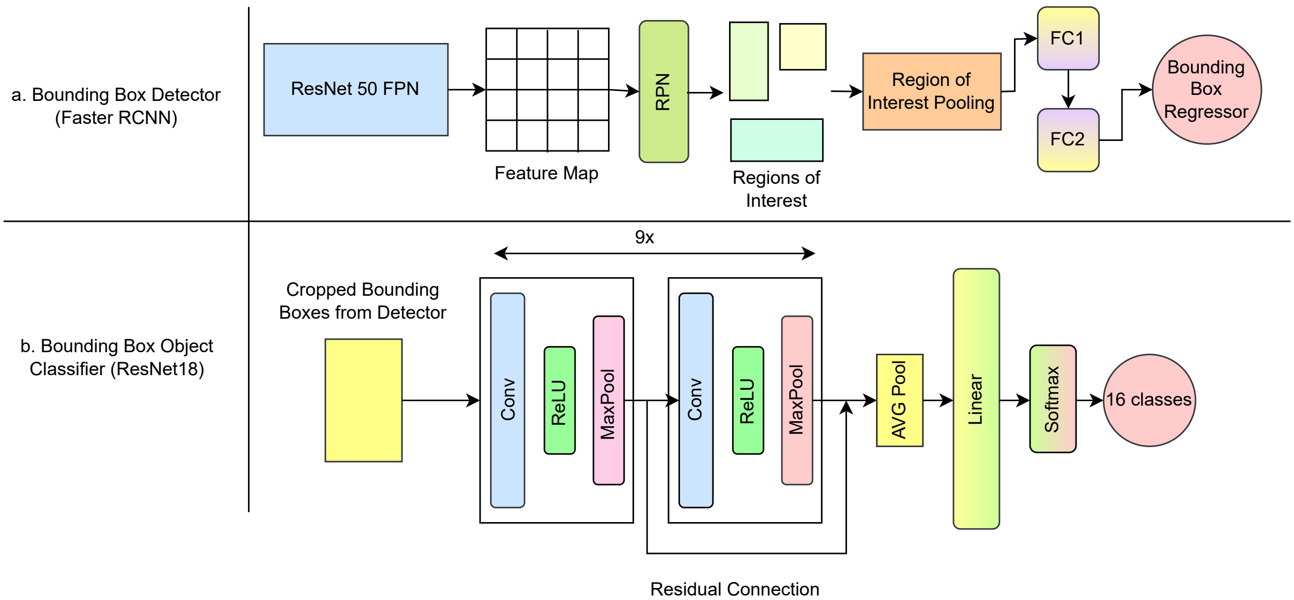

Figure 1: Figure (a) demonstrates the detector (Faster RCNN) that localizes objects in image collecting features using an FPN backbone and proposing objects using the RPN network. The model is trained to correctly collect features and propose regions only as its classification head is removed. The regions predicted by this model are cropped and fed to a classifier (ResNet18) to predict object classes.

2.1 Model

We utilize two different models for localization and classification. For localization, we utilize the Faster RCNN model with a ResNet50 FPN, and for classification, we use the ResNet18 model.

Faster RCNN ResNet50 FPN: The Faster RCNN model can both classify and localize objects. When dealing with an image, the Faster RCNN collects important features from an image using its backbone to get feature maps. The backbone is usually a large model to learn many features altogether. The feature map is sent to the Region Proposal Network (RPN). The RPN treats every pixel in the feature map as an anchor point and generates several boxes of different shapes, scales, and aspect ratios, also known as anchor boxes. The anchor boxes are proposed regions that might contain objects using small convolutions. The RPN calculates a tuned objectness score to understand whether a box contains an object. These anchor box regions are fed to a Fast RCNN detector that pools these regions of interest, extracts feature using the same backbone and passes onto two different fully connected networks to predict the object class and the bounding box. The complete model is trained using a multi-task loss function given as:

$$L(p_i, t_i, v_i) = \frac{1}{N_{cls}} \sum COE(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum p_i L1(t_i, v_i) \tag{1}$$

Here, in equation 1 N_{cls} and N_{reg} is the number of regions of interest used for the classification and detection bounding box. p_i and v_i are the predicted classification probability and bounding box for the i-th region, respectively. p_i^* and t_i are the classifications and bounding box ground truth. COE and $L1$ represent the Cross-Entropy and Mean Absolute Error Loss. We trained the model using the train set and evaluated the model on the validation set. After training the model using the train set, we remove the classification layer so that the model has no parameters that are used for classification. Now, the model can only predict bounding boxes. We trained the model for 7 epochs.

ResNet18: We used the ResNet18 architecture to classify the objects in the predicted bounding box by the Faster RCNN model. The ResNet18 model consists of 18 deep layers with several convolutional layers, enabling it to capture several important features. The ResNet model utilizes residual connections that jump across one or more layers to provide shortcuts between them. The primary problem of vanishing gradient that deep networks encounter was intended to be addressed by introducing these shortcut links. We trained the ResNet model using images of objects cropped according to the ground truth bounding boxes of the train set and, tested the model on the ground truth bounding boxes cropped of the validation set and trained it for 100 epochs.

End-to-End System: The Faster RCNN model without its classification head generates Regions of interest, which is passed onto to the ResNet18 model to predict class labels that are used to classify regions and remove unwanted regions.

2.2 Model Compression

Deep learning models usually require many parameters to capture important and necessary features for predictions. Model compression techniques allow deep-learning models to produce lightweight models without compromising performance to a certain extent.

Teacher-Free Knowledge Distillation: Knowledge distillation Hinton et al. (2015) is a model compression technique where a teacher model imparts knowledge to the student model, allowing the students to learn from the teacher directly in addition to the dataset through a customized loss, distillation loss 3, that aligns the output of the student logits to the teacher logits using the Kullback Liebler Divergence function 2 as $\mathcal{L}_{KLD}$ unlike normal training that only utilizes cross entropy loss 2 as $\mathcal{L}_{COE}$. This allows the smaller inept student model to attain higher performance, which can then be deployed to mobile or edge devices.

$$\mathcal{L}_{CE}(y, h_s) = -\sum_{c=1}^{n} y(c) \log h_s(c) \qquad \mathcal{L}_{KLD}(o_s, o_m) = \tau^2 \sum_{c=1}^{n} o_m(c) \log \frac{o_s(c)}{o_m(c)} \tag{2}$$

$$\mathcal{L} = \alpha\ \mathcal{L}_{KLD}(o_s, o_m) + (1-\alpha)\ \mathcal{L}_{CE}(y, h_s) \tag{3}$$

Here, y is the ground truth, and o_s and o_m are the student and teacher logits. h_s is the student logits before softening with τ, temperature, a hyperparameter. However, the process is computationally inefficient, requiring a trained and accurate teacher model for student training. The teacher model has to be inferred to train the student model during training, which makes the procedure computationally heavy. To tackle this problem, we use teacher-free knowledge distillation Yuan et al. (2020), where the teacher logits are synthetically generated. The ground truth labels from the dataset are used to generate teacher logits where the true class has a 99%. This allows replicating a teacher's best possible output without being exactly similar to the labels. As the dataset is imbalanced, we use Focal Loss instead of Cross Entropy Loss in the Ditillation Loss function. This is given by:

$$\mathcal{L} = \alpha\ \mathcal{L}_{KLD}(o_s, o_m) + (1-\alpha)\ \mathcal{L}_{Foc}(y, h_s) \tag{4}$$

Post-Training Quantization: Quantization methods reduce parameter precision in models, allowing faster inference and a smaller model size. We performed post-training quantization on the classification model to quantify 32-bit floating point weights to 8-bit integer precision. This is done by calculating two parameters, scale and value, that are used to quantize both inputs and the weights of the models and later dequantize to get the actual output.

Deployment We deploy the end to end model to our local system. The system offers a friendly interface that allows users to upload pictures and the model returns correctly predicted bounding boxes and proper classifications of objects in the bounding box. We showcase our system in the Appendix.

3 Data Set

In this study, we investigate the Teledyne FLIR dataset to teach computer vision models to determine objects in thermal imagery effectively. The Teledyne FLIR dataset consists of 26,442 annotated RGBT (Red, Green, Blue, Thermal) images collected through a thermal and visible camera pair mounted on a vehicle. The dataset consists of 9711 thermal images and 9233 RGB images split into train and validation sets at a 90%-10% ratio. Furthermore, for testing, a pair of 3749 thermal/RGB videos are also presented to be utilized. The videos are captured at 30 FPS and consists of 7498 frames. The dataset covers 15 different common objects found in streets that may alter the movement of automated systems or vehicles. Among which the most popular in both the RGB and thermal sets are person (Thermal: ≈ 55000, RGB: ≈ 38000), car (Thermal: ≈ 80000, RGB: ≈ 78000) and sign (Thermal: ≈ 22000, RGB: ≈ 34000). Visible/RGB imagery in such scenarios where there is noise results in improper localization and classification of objects due to noises captured in RGB images, such as bad lighting, glares, fog, etc. Thermal imaging captures radiation emitted by objects, eliminating noise. We provide a detailed visualization in figures: 2, 3, 4 of the used dataset to show the examples and the efficacy of thermal images over RGB images.

Figure 2: Glare can conceal important objects and dangerous situations. Left: RGB image, Right: Thermal Image

Figure 3: Lack of light hampering visual and concealing objects. Left: RGB image, Right: Thermal Image

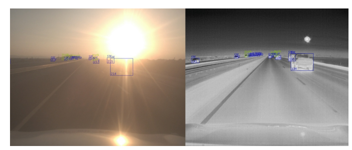

Figure 4: Weather conditions may produce noise in RGB images. Left: RGB image, Right: Thermal Image

Figure 5: Example images from the dataset.

4 Experiment and Results

This section showcases the results obtained in this study. In this study, we carry out several different experiments using different models. This section showcases our findings. The performance metrics are detailed in the appendix below.

4.1 Classification

We perform the classification task using the ResNet18 model to predict objects inside the bounding boxes generated by the localization model. We crop bounding boxes in the train and validation set to use as training and testing data for the model. We further collect negative samples so that the ResNet18 model can remove the negative anchors. We train the ResNet18 model using teacher-free knowledge distillation (TF-KD) and the traditional procedure. We further utilize focal loss instead of Cross-Entropy in TF-KD to check its effect on the imbalanced data. Table 1 showcases the results obtained. It can be established that the models trained on Tf-KD outperform the model trained on Cross-Entropy by 3 and 4% in accuracy and also in precision, recall, and F1-score. This is due to synthetic logits used as additional knowledge for the ResNet model. The model that uses Focal Loss to learn the dataset instead of Cross-Entropy Loss in TF-KD performs better than the latter. This is due to focal loss addressing imbalanced scenarios by adding a modulating term to Cross-Entropy Loss. Both the models are then quantized to make them 8 times smaller and faster, shown in Table 2 without much loss in performance. The model trained using Tf-KD with COE further shows imbalance issues as quantization causes a major loss in performance of about 3% whereas the other model only suffers by 0.02%.

Training	Loss Function	Accuracy	Precision	Recall	F1 Score
Normal	COE	87.31%	89.44	86.23	87.10
Tf-KD	COE+KLD	90.41	90.41	90.08	90.08
Tf-KD	COE+Focal Loss	91.10%	90.54	91.10	90.65

Table 1: Performance of ResNet models utilizing different training techniques and loss functions trained and evaluated on the cropped objects in the ground truth bounding boxes of the dataset.

Loss	Accuracy	Model Size (MB)	Validation Inference Time	Quantized Accuracy	Quantized Size (MB)	Validation Inference Time(Quantized)
COE+KLD	90.41%	87.47	13.57s	87.69%	10.95	9.21s
COE+Focal Loss	91.10%	87.47	13.4s	91.08%	10.95	9.32s

Table 2: Efficacy of different Tf-KD trained ResNet18 model after post-training quantization.

Detection: For the localizer, we trained a FasterRCNN model as a baseline on the Teledyne FLIR thermal dataset. The performance of the model, shown in Table 2, has an mAP (Mean Average Precision) at IoU = 0.5, of 19.16% after 7 epochs. We further trained another Faster RCNN model only to predict negative and positive boxes and the bounding box. We removed the classification head of this model and utilized the ResNet18 model trained on TF-KD (Focal Loss) as a classifier for all classes and to remove negative anchors. The model achieved a performance of 22.96% mAP @ IoU = 0.5, scoring high AP% on detecting the most common objects in the streets: Car (67.38%) and Person (57.47%). However, we realized the classification ResNet18 model was not great at detecting negative boxes. So we allowed the classification head to remain to remove negative boxes. This allowed the mAP @ IoU = 0.5 to increase by 2.5% and increased person and car AP% by approximately 2% each. We also showcase

Model	mAP @IoU = 0.5	Person AP	Car AP
Faster RCNN	19.16%	56.31%	39.11%
Faster RCNN + ResNet18	22.96%	67.38%	57.47%
Faster RCNN (wth class-head) + ResNet18	25.38%	68.57%	59.69%

Table 3: Performance of Faster RCNN with different architectural manipulations as basic Faster RCNN, Faster RCNN with ResNet18 classifier, and Faster RCNN with ResNet18 classifier and internal negative anchor remover.

Figures 6,7,8 show the same images. Figure 7 showcases the ground truth. Figure 8 is the generation with a Faster RCNN without a classifying head. As a result, it cannot remove the unnecessary anchor boxes. Figure 9 The Faster RCNN uses a ResNet classifying head. As a result, the outputs are accurate.

Figure 6: Image Ground Truth: labels and box

Figure 7: Faster RCNN generation without classifier

Figure 8: Faster RCNN generation with classifier

5 Conclusion

In this study, we started with the SSD model. However, the SSD model was unable to properly learn the FLIR dataset. As a result we used the Faster RCNN model. The Faster RCNN model is an older model. As a result, it does not use the state of the art techniques. As a result, its performance is a bit lackluster. The Faster RCNN with a large classifier works better than a basic Faster RCNN. However, the performance increase is minute compared to the increase in the number of parameters. As a result, we looked to apply model compression to our system to enable a faster and more efficient model. However, model compression for the localizer (Faster RCNN) did not work as well as for the classifier. Quantizing the Faster RCNN model causes a large loss in its performance. As a result, we were unable to quantize the Faster RCNN model. We used the ResNet model as it is not prone to vanishing gradient problems. Quantizing the ResNet model was difficult as it had many layers. However, we were able to use the Open-Vino library to quantize and enhance model performance. Along with that, training these models was a big issue in this project due to lack of available computing resources. We used Google colab to train the models. The models used were not tuned on their hyper-parameters due to the unavailability of computing resources. For future work, we look to use the latest models. We will also look to use smaller and efficient model for usage in realtime purposes.

References

LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *nature* **2015**, *521*, 436–444.

Girshick, R. Fast r-cnn. Proceedings of the IEEE international conference on computer vision. 2015; pp 1440–1448.

He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. Proceedings of the IEEE international conference on computer vision. 2017; pp 2961–2969.

Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A. C. Ssd: Single shot multibox detector. Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14. 2016; pp 21–37.

Redmon, J.; Farhadi, A. YOLO9000: better, faster, stronger. Proceedings of the IEEE conference on computer vision and pattern recognition. 2017; pp 7263–7271.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal loss for dense object detection. Proceedings of the IEEE international conference on computer vision. 2017; pp 2980–2988.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C. L. Microsoft coco: Common objects in context. Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13. 2014; pp 740–755.

Everingham, M.; Van Gool, L.; Williams, C. K.; Winn, J.; Zisserman, A. The pascal visual object classes (voc) challenge. *International journal of computer vision* **2010**, *88*, 303–338.

Wu, Y.; Kirillov, A.; Massa, F.; Lo, W.-Y.; Girshick, R. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

Hinton, G.; Vinyals, O.; Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* **2015**,

Yuan, L.; Tay, F. E.; Li, G.; Wang, T.; Feng, J. Revisiting knowledge distillation via label smoothing regularization. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020; pp 3903–3911.

6 Appendix

Deployment System: We deployed the end-to-end model locally. We designed a user-friendly GUI (Graphics User Interface) that allows users to easily select image files from any directory and upload them so that the model can detect objects and predict their classes. The system also allows removing the images and uploading new images and allows the user to flag outputs that are not up to expectations. Figure 9 shows an example of our system being used.

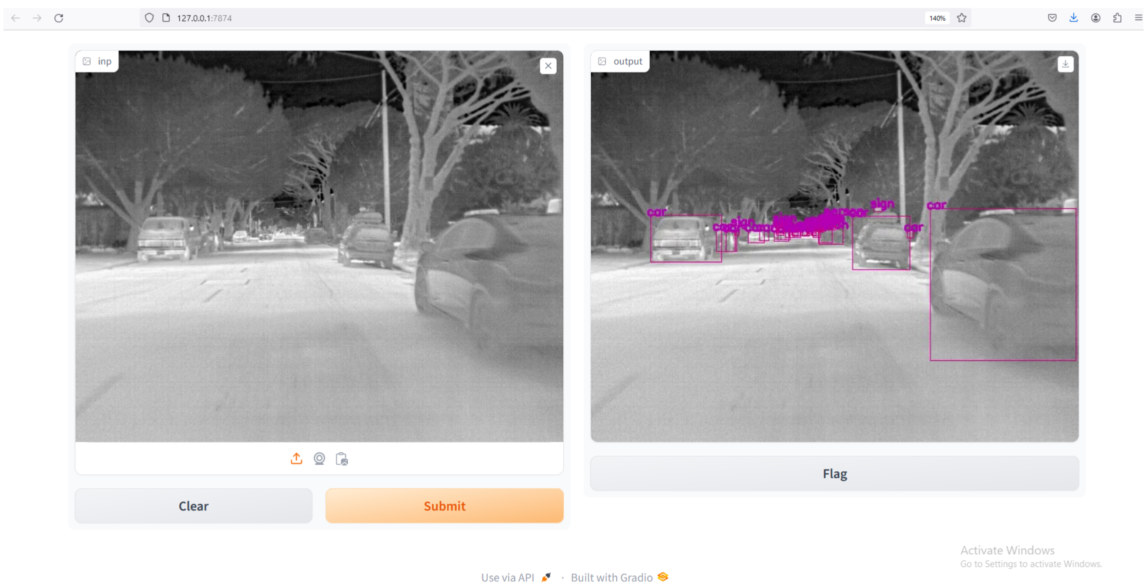

Figure 9: The deployed system using our end-to-end Faster RCNN localizer with ResNet18 classifier on a thermal image example.

Accuracy: The percentage of correctly categorized data instances relative to the total number of data instances is known as accuracy. Stated differently, it is the ratio of properly predicted labels to the number of active labels.

Precision: The precision metric quantifies the proportion of accurate predictions generated by the model. In other words, it displays the frequency with which an ML model predicts the target class correctly.

Recall: A classification model's recall is its capacity to recognize each and every data point in a pertinent class.

F1 Score: The harmonic mean of a classification model's accuracy and recall is known as the F1 score. The F1 score accurately reflects a model's dependability since both metrics (precision and recall) have an equal weight in the score. A F1 score falls between 0 and 1. A performance score of 0 denotes worst possible performance, whereas a score of 1 denotes best possible performance.

Confusion Matrix: An overview of a machine learning model's performance on a set of test data is provided via a confusion matrix. It demonstrates the quantity of false positives, false negatives, true positives, and true negatives. It is frequently used to assess the performance of classification models that aim to assign a categorical label to each input instance. The confusion matrix serves the crucial function of highlighting the model's quality when the classes are unbalanced.

IoU: Intersection over Union (IoU) is a common metric used to calculate localization errors and assess localization accuracy in object detection models. It determines how much two bounding boxes (a predicted bounding box and a ground truth bounding box) overlap. The IoU magnitude increases with the size of the overlap region. A score of 0 denotes no overlap between the boxes, while a score of 1 denotes a perfect overlap between the predicted box and the ground truth box.

mAP: Object detection models such as R-CNN and YOLO are evaluated using the mean average precision, or mAP. After comparing the detected box with the ground-truth bounding box, the mAP generates a score. The more precise the model's detections, the higher the score.